Word2vec vector analysis

March 10, 2021

1 SI630 Homework 2: Word2vec Vector Analysis

Important Note: Start this notebook only after you've gotten your word2vec model up and running!

Many NLP packages support working with word embeddings. In this notebook you can work through the various problems assigned in Task 3. We've provided the basic functionality for loading word vectors using Gensim, a good library for learning and using word vectors, and for working with the vectors.

One of the fun parts of word vectors is getting a sense of what they learned. Feel free to explore the vectors here!

[1]:
```
from gensim.models import KeyedVectors
from gensim.test.utils import datapath
```

I am HuJiaoyang on Kaggle.

Below is the results from the non-synonym-aware model.

[2]:
```
word_vectors_no_syn = KeyedVectors.
↪load_word2vec_format('gensim_KeyedVectors_no_syn_final.txt', binary=False)
```

[3]:
```
word_vectors_no_syn.similar_by_word("books")
```

[3]:
```
[('articles', 0.8948364853858948),
 ('novels', 0.892451286315918),
 ('essays', 0.8862031102180481),
 ('poems', 0.8730127811431885),
 ('illustrations', 0.8594130277633667),
 ('stories', 0.8585752248764038),
 ('publications', 0.8579347729682922),
 ('entries', 0.8464248180389404),
 ('volumes', 0.8440317511558533),
 ('columns', 0.8392201662063599)]
```

That is great results from books! We can safely say that all the 10 entries are book and literature related.

[4]:
```
word_vectors_no_syn.similar_by_word("good")
```

[4]:
```
[('bad', 0.8490206599235535),
 ('happy', 0.822114884853363),
 ('bigger', 0.8186966776847839),
 ('decent', 0.8155100345611572),
 ('little', 0.8149300813674927),
 ('remarkable', 0.8137297630310059),
 ('poor', 0.807860255241394),
 ('tough', 0.8057727217674255),
 ('better', 0.8027393817901611),
 ('brilliant', 0.7974538803100586)]
```

We see that the closet word to is in fact , this makes total sense as good and bad have basically the same usage, except that the meaning is reversed.

[5]:
```
word_vectors_no_syn.similar_by_word("man")
```

[5]:
```
[('person', 0.8697253465652466),
 ('woman', 0.8140915632247925),
 ('spider', 0.7498606443405151),
 ('killing', 0.7486321926116943),
 ('men', 0.735857367515564),
 ('fantastic', 0.7293199896812439),
 ('lonely', 0.7272354364395142),
 ('unsung', 0.7257636785507202),
 ('sacrifice', 0.7256767749786377),
 ('samurai', 0.7229107618331909)]
```

We see person, woman, men which means that our model is successful.

[6]:
```
word_vectors_no_syn.similar_by_word("i")
```

[6]:
```
[('know', 0.8168253302574158),
 ('you', 0.7998055219650269),
 ('guess', 0.7970436811447144),
 ('think', 0.7909340262413025),
 ('we', 0.7880868911743164),
 ('want', 0.786977231502533),
 ('forget', 0.7781956791877747),
 ('gonna', 0.7688604593276978),
 ('maybe', 0.7611671686172485),
 ('if', 0.7593001127243042)]
```

[7]:
```
word_vectors_no_syn.similar_by_word("beautiful")
```

[7]:
```
[('wonderful', 0.8825747966766357),
 ('happiness', 0.8565652966499329),
 ('amazing', 0.8417251110076904),
 ('funny', 0.8397625684738159),
 ('precious', 0.8370656967163086),
```

```
 ('lonely', 0.8336613178253174),
 ('forever', 0.8330222368240356),
 ('hearted', 0.8318878412246704),
 ('loving', 0.8314868211746216),
 ('lovely', 0.8295010924339294)]
```

[8]:
```
word_vectors_no_syn.similar_by_word("hotel")
```

[8]:
```
[('cottage', 0.8615527153015137),
 ('restaurant', 0.8587293028831482),
 ('garden', 0.8349941968917847),
 ('ranch', 0.8252112865447998),
 ('plantation', 0.8203280568122864),
 ('park', 0.8184698820114136),
 ('mansion', 0.8145043849945068),
 ('resort', 0.8074013590812683),
 ('castle', 0.805768609046936),
 ('casino', 0.8002871870994568)]
```

I consider all the words above to be common words, and indeed we see great performance in all of them. Nearly every single generated words make sense, and have similar meaning or similar usage with our target word.

[9]:
```
word_vectors_no_syn.similar_by_word("elephant")
```

[9]:
```
[('arrow', 0.9130080938339233),
 ('infinity', 0.9093651175498962),
 ('lonesome', 0.8933923244476318),
 ('phantom', 0.8873546123504639),
 ('monkey', 0.8861550092697144),
 ('pig', 0.8858698606491089),
 ('madness', 0.8853882551193237),
 ('thief', 0.8806638717651367),
 ('robots', 0.8806427717208862),
 ('rainbow', 0.8779715895652771)]
```

Elephant is an occasional word, we find that unlike the common words above, which seems like every single of the 10 similar words are in fact semantically similar, the 10 words our model generate for elephant is a little bit off. But we can still see monkey and pig, which are also animals.

[10]:
```
word_vectors_no_syn.similar_by_word("arctic")
```

[10]:
```
[('antarctic', 0.9053131341934204),
 ('polar', 0.8864127397537231),
 ('potato', 0.8832401037216187),
 ('goldfields', 0.8799384832382202),
 ('shipbuilding', 0.8658095598220825),
 ('siberian', 0.8616619110107422),
```

```
 ('connecting', 0.8600573539733887),
 ('scandinavian', 0.8598737120628357),
 ('explorers', 0.857343852519989),
 ('meteorological', 0.8560659885406494)]
```

Arctic is also an occasional, if not rare word. But we can still see related words like antarctic, polar, siberian, explorers.

[11]:
```
word_vectors_no_syn.similar_by_word("neuron")
```

[11]:
```
[('topology', 0.8367322683334351),
 ('bacterial', 0.81732177734375),
 ('seismic', 0.8128526210784912),
 ('quantitative', 0.7918833494186401),
 ('lithium', 0.7785747051239014),
 ('lebedev', 0.7741100788116455),
 ('kinetic', 0.7714471220970154),
 ('discrete', 0.7709615230560303),
 ('sickle', 0.7707749605178833),
 ('optimization', 0.7684822678565979)]
```

Neuron is no doubt a rare word, but we can see biology related terminology, like bacterial, and all words are science-related in general.

[12]:
```
word_vectors_no_syn.similar_by_word("anaconda")
```

[12]:
```
[('showroom', 0.8053072690963745),
 ('vu', 0.8048404455184937),
 ('masjid', 0.8036600351333618),
 ('tuscan', 0.8003712892532349),
 ('sidewalk', 0.7966856956481934),
 ('chop', 0.7956990003585815),
 ('rundfunk', 0.7937109470367432),
 ('sunnyside', 0.7930664420127869),
 ('foreground', 0.7926075458526611),
 ('jaish', 0.7893842458724976)]
```

Anaconda is no double a rare word, and no word generated here makes any sense. We can conclude that very little text about anaconda is in our wiki-bios.med.txt, which lead to the poor performance.

[13]:
```
def get_analogy_no_syn(a, b, c):
    return word_vectors_no_syn.most_similar(positive=[b, c], negative=[a])[0][0]
```

[14]:
```
get_analogy_no_syn('man', 'woman', 'king')
```

[14]:
```
'queen'
```

[15]:
```
get_analogy_no_syn('man', 'woman', 'actor')
```

[15]:
```
'actress'
```

[16]:
```
get_analogy_no_syn('father', 'mother', 'uncle')
```

[16]:
```
'aunt'
```

[17]:
```
get_analogy_no_syn('old', 'new','ancient')
```

[17]:
```
'modern'
```

[18]:
```
get_analogy_no_syn('japan', 'korea', 'tokyo')
```

[18]:
```
'seoul'
```

From the examples above we can see that our model can comprehend the analogies tested.

Up next we will test the synonym-aware model.

[20]:
```
word_vectors_syn = KeyedVectors.load_word2vec_format('gensim_KeyedVectors.txt',␣
↪binary=False)
```

[21]:
```
word_vectors_syn.similar_by_word("books")
```

[21]:
```
[('novellas', 0.8433401584625244),
 ('novels', 0.822393000125885),
 ('articles', 0.8132258653640747),
 ('essays', 0.8005460500717163),
 ('editions', 0.7978662252426147),
 ('cookbooks', 0.7810624837875366),
 ('translations', 0.7802404165267944),
 ('monographs', 0.7750540375709534),
 ('photographs', 0.7576437592506409),
 ('publications', 0.7565302848815918)]
```

[22]:
```
word_vectors_syn.similar_by_word("good")
```

[22]:
```
[('practiced', 0.7863052487373352),
 ('commodity', 0.7799349427223206),
 ('expert', 0.7776026129722595),
 ('goodness', 0.7633113861083984),
 ('tanning', 0.7354745864868164),
 ('adept', 0.7298815250396729),
 ('1', 0.7278332114219666),
 ('proficient', 0.715816855430603),
 ('serviced', 0.7099287509918213),
 ('skillful', 0.7073875665664673)]
```

[23]:
```
word_vectors_syn.similar_by_word("man")
```

[23]:
```
[('serviceman', 0.8590536117553711),
 ('human', 0.8466517925262451),
 ('homo', 0.8445079326629639),
 ('natural', 0.7502440214157104),
 ('autism', 0.7372031211853027),
 ('occupational', 0.7313769459724426),
 ('contraceptive', 0.7310788631439209),
 ('solving', 0.7281184196472168),
 ('holistic', 0.7275974154472351),
 ('reproductive', 0.7134917974472046)]
```

[24]:
```
word_vectors_syn.similar_by_word("i")
```

[24]:
```
[('unrivaled', 0.8721108436584473),
 ('1', 0.8398638963699341),
 ('peerless', 0.8386151790618896),
 ('unmatched', 0.8383300304412842),
 ('matchless', 0.8290156722068787),
 ('unrivalled', 0.813519299030304),
 ('ane', 0.811320424079895),
 ('one', 0.8043263554573059),
 ('odd', 0.7935372591018677),
 ('oneness', 0.717767059803009)]
```

[25]:
```
word_vectors_syn.similar_by_word("beautiful")
```

[25]:
```
[('romping', 0.785847008228302),
 ('lovin', 0.7839919328689575),
 ('darker', 0.7720255851745605),
 ('lullabies', 0.7639796733856201),
 ('screams', 0.7634345293045044),
 ('perverted', 0.7567563056945801),
 ('undeniable', 0.754807710647583),
 ('evermore', 0.7545579671859741),
 ('playfulness', 0.7515984773635864),
 ('wildest', 0.7499814629554749)]
```

[26]:
```
word_vectors_syn.similar_by_word("hotel")
```

[26]:
```
[('eatery', 0.7699471712112427),
 ('refinery', 0.7471675872802734),
 ('waterfront', 0.7419823408126831),
 ('cassino', 0.7361522316932678),
 ('store', 0.7291527986526489),
 ('casino', 0.7272229790687561),
 ('docks', 0.7268884181976318),
 ('restaurant', 0.723502516746521),
 ('coliseum', 0.7190192937850952),
```

```
 ('malibu', 0.718827486038208)]
```

we can see that with synonyms added, bad is no longer in the closest 10 words of good. The closest 10 words for both good and man all seem to be worse than before.

[27]:
```
word_vectors_syn.similar_by_word("elephant")
```

[27]:
```
[('snare', 0.8062669634819031),
 ('feathered', 0.8036626577377319),
 ('harajuku', 0.8034716844558716),
 ('flaming', 0.8015307188034058),
 ('pies', 0.7986346483230591),
 ('shadows', 0.7984472513198853),
 ('ophidian', 0.7964584231376648),
 ('goose', 0.7902676463127136),
 ('ghost', 0.7887022495269775),
 ('unicorn', 0.7862536907196045)]
```

[28]:
```
word_vectors_syn.similar_by_word("arctic")
```

[28]:
```
[('ocean', 0.7639082670211792),
 ('navigation', 0.7459009885787964),
 ('siberian', 0.7451314926147461),
 ('geodesic', 0.7384567260742188),
 ('antarctic', 0.7327370047569275),
 ('alaskan', 0.7162798047065735),
 ('spacelab', 0.7151463627815247),
 ('oceans', 0.7135401964187622),
 ('caspian', 0.7133638858795166),
 ('himalayan', 0.7122912406921387)]
```

The occasional word elephant is worse, as there are no more animals except goose in the 10 words. Arctic is still good, as we can see ocean, siberian, antarctic, alaskan etc.

[29]:
```
word_vectors_syn.similar_by_word("neuron")
```

[29]:
```
[('seismic', 0.8791794180870056),
 ('quantitative', 0.8590160012245178),
 ('plutonium', 0.8571957349777222),
 ('magnetism', 0.8504460453987122),
 ('hydrodynamics', 0.8496783971786499),
 ('qualitative', 0.8477330803871155),
 ('bacterial', 0.8472901582717896),
 ('isotope', 0.8441588878631592),
 ('dynamics', 0.8437991142272949),
 ('diagnostics', 0.8422766923904419)]
```

[30]:
```
word_vectors_syn.similar_by_word("anaconda")
```

[30]:
```
[('feathered', 0.7986303567886353),
 ('dumped', 0.7952090501785278),
 ('trilateral', 0.7830404043197632),
 ('bulb', 0.7814380526542664),
 ('deadliest', 0.7782955765724182),
 ('powerbomb', 0.7773516178131104),
 ('triangle', 0.7704331874847412),
 ('robot', 0.7688275575637817),
 ('xa', 0.76659095287323),
 ('recon', 0.7658909559249878)]
```

The words related to neuron are very close, and perhaps better than the non-synonym-aware model. The anaconda is also slightly better, as we can see deadliest in the words, but overall still pretty bad due to the lack of information regarding anaconda in our training data.

[33]:
```
def get_analogy_syn(a, b, c):
    return word_vectors_syn.most_similar(positive=[b, c], negative=[a])[0][0]
```

[34]:
```
print(get_analogy_syn('man', 'woman', 'king'),
get_analogy_syn('man', 'woman', 'actor'),
get_analogy_syn('father', 'mother', 'uncle'),
get_analogy_syn('old', 'new','ancient'),
get_analogy_syn('japan', 'korea', 'tokyo'))
```

```
hele singer grandpa chivalric angamaly
```

Curiously the addition of synonyms seems to complete screw up my analogies. The first four are all incorrect. Pyeongchang is the capital of North Korea so I count it as correct.

Now, we check some words that are in the synonyms.txt

[39]:
```
print(word_vectors_no_syn.similar_by_word("barrage"))
print(word_vectors_syn.similar_by_word("barrage"))
```

```
[('flurry', 0.8900510668754578), ('transcription', 0.8856087923049927),
('recurrence', 0.8820905685424805), ('accumulation', 0.8809360265731812),
('intensity', 0.8713709115982056), ('deterioration', 0.8712615966796875),
('fraction', 0.8698258399963379), ('humiliation', 0.8696500062942505),
('backdrop', 0.8693416714668274), ('flock', 0.8677707314491272)]
[('shelling', 0.8384448289871216), ('detachment', 0.8257587552070618), ('onset',
0.8215346336364746), ('assault', 0.818076491355896), ('bombardment',
0.8138667941093445), ('artillery', 0.8107582926750183), ('onslaught',
0.7990705966949463), ('battery', 0.7814236879348755), ('rape',
0.7759183645248413), ('ammunition', 0.7736815214157104)]
```

We can see that with the addition of synonyms, the weights of the synonyms of barrage,battery bombardment shelling, have significantly increased.

In general, I prefer the non-synonym-aware model as it usually produces better analogies and similarities(at least with my implementation), while the addition of synonyms can indeed increase

the weights of occasional words and rare words, the cost seems to be great, as we see in our testing that my word analogies with the synonym-aware model becomes completely screwed.